(19) (11) Publication number: 08030558 A

Generated Document.

PATENT ABSTRACTS OF JAPAN

(21) Application number: 06167902 (51) Intl. Cl.: G06F 15/16

(22) Application date: 20.07.94

(30) Priority:

(43) Date of application publication: 02.02.96

(84) Designated contracting states:

(71) Applicant: FUJITSU LTD

(72) Inventor: UKAI TAKANORI
UEDA HARUYASU

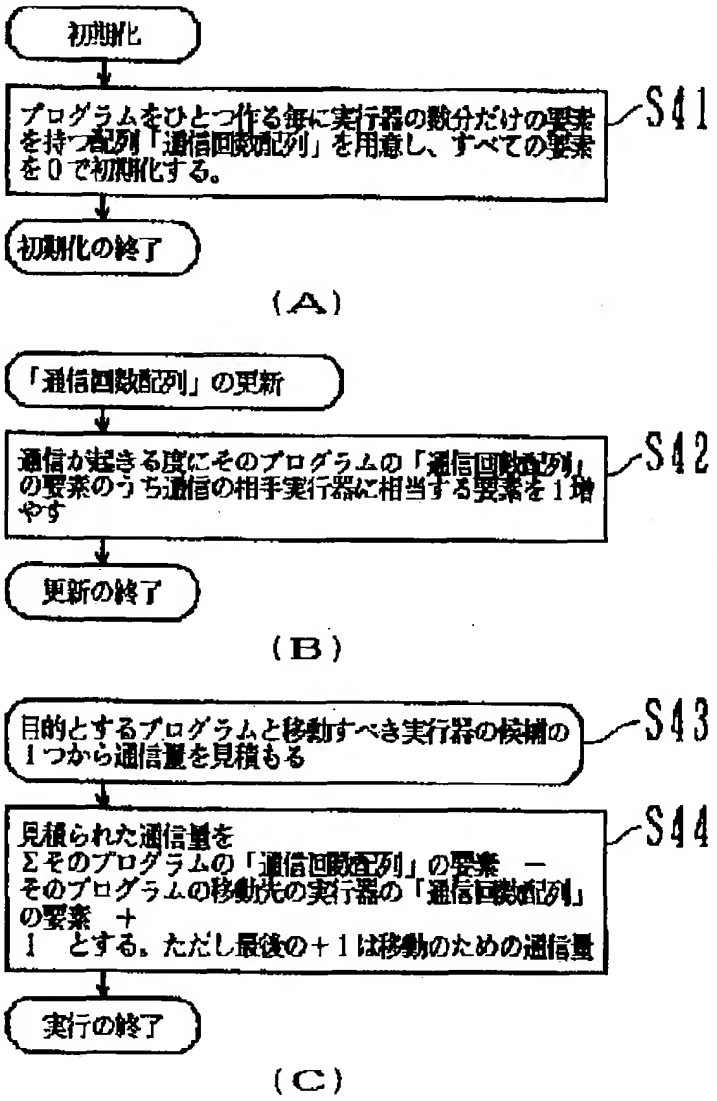
(74) Representative:

(54) LOAD DISTRIBUTING METHOD IN COMPUTER SYSTEM AND COMPUTER SYSTEM UTILIZING THE METHOD

(57) Abstract:

PURPOSE: To improve the processing efficiency of the whole computer system by properly distributing load.

CONSTITUTION: An execution device (i) adds '1' to the value of a corresponding program element in a communication frequency array in each communication of a program in execution with each execution device (S42). The volume of communication between a program (m) to be a moving candidate and an execution device (n) to be a candidate to which the program (m) is to be moved is estimated (S43). Communication frequency with the execution device (n) is subtracted from the sum of values of respective arrays in the program (m) to be the moving candidate of the communication frequency array, i.e., the sum of communication frequency, '1' is added to the substracted result and the added result is set up as the volume of communication after moving the program (m). On the other hand, the sum of elements of an execution device other than the execution device (i) is set up as the volume of communication obtained before the movement of the program (m). Then the communication volume before movement and the communication volume after movement are respectively multiplied by a communication time coefficient '1' to find out communication time before movement and communication time after movement, and when a combination between a program and an execution device by which the communication time after movement is shorter than the communication time before movement exists, the program is allocated to the execution device.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報**（Ａ）　(11)特許出願公開番号

特開平8－30558

(43)公開日　平成８年(1996)２月２日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ　15/16	３７０　Ｎ			

審査請求　未請求　請求項の数６　ＯＬ　（全８頁）

(21)出願番号　特願平6－167902

(22)出願日　平成６年(1994)７月20日

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中1015番地
(72)発明者　鵜飼　孝典
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内
(72)発明者　上田　晴康
神奈川県川崎市中原区上小田中1015番地
富士通株式会社内
(74)代理人　弁理士　大菅　義之　（外１名）

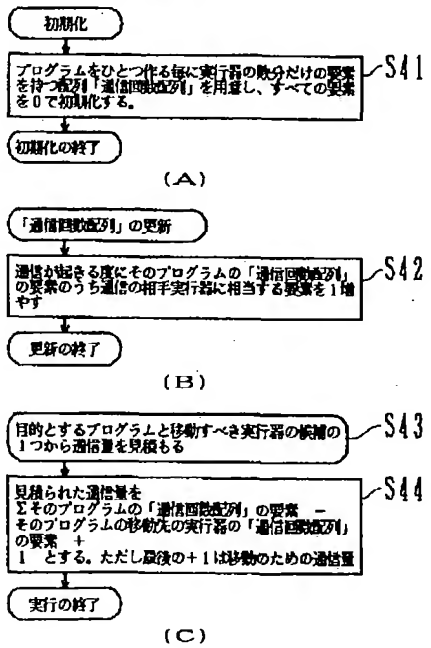
(54)【発明の名称】　計算機システムにおける負荷分散方法及びそれを利用した計算機システム

(57)【要約】

【目的】　本発明の目的は、負荷を適正に分配することにより計算機システム全体の処理効率を向上させることである。

【構成】　実行器ｉは、実行中のプログラムが各実行器と通信を行う度に、通信回数配列の該当するプログラムの要素の値を「１」インクリメントする（図６（Ｂ）のＳ４２）。移動候補のプログラムｍとそのプログラムｍの移動先候補の実行器ｎとの間の通信量を見積もる（同図（Ｃ）のＳ４３）。そして、通信回数配列の移動候補のプログラムｍの各配列の値の総和、即ち通信回数の総和から、移動先候補の実行器ｎとの通信回数を減算し、減算結果に「１」を加えたものを、プログラムｍの移動後通信量とする。また、実行器ｉ以外の他の実行器の要素の総和をプログラムｍの移動前通信量とする。そして、移動前通信量及び移動後通信量に通信時間係数として「１」を乗算して移動前通信時間及び移動通信時間を求め、移動後通信時間が移動前通信時間より短くなるプログラムと実行器の組み合わせが存在したなら、その実行器にプログラムを割り当てる。

通信量見積り処理のフローチャート



1

【特許請求の範囲】

【請求項１】　複数のプログラムをそれぞれ実行する複数の実行器と、該複数の実行器を制御する制御器とからなる計算機システムにおいて、

前記実行器が、

実行すべきプログラムと自己の実行器を含む各実行器との間の通信量を見積もる通信量見積り手段と、

前記通信量見積り手段によって見積もられた通信量に基づいて、前記プログラムを自己の実行器で実行した場合の該プログラムと他の実行器との間の移動前通信時間と、該プログラムを他の実行器に移動させた場合の該プログラムと移動先実行器以外の他の実行器との間の移動後通信時間とを算出する通信時間算出手段と、

前記通信時間算出手段によって算出された移動後通信時間が移動前通信時間より短くなるプログラムと実行器との組み合わせを抽出し、該抽出した実行器に該プログラムを割り当てるように前記制御器に通知する負荷割り当て手段とを備えることを特徴とする計算機システム。

【請求項２】　前記通信量見積り手段は、前記プログラムと自己の実行器を含む各実行器との間の過去の通信量を記憶する通信量記憶手段を有し、

前記通信時間算出手段は、前記記憶手段に記憶されている過去の通信量から前記移動前通信時間と前記移動後通信時間を算出することを特徴とする請求項１記載の計算機システム。

【請求項３】　前記通信量見積り手段は、前記プログラムと前記各実行器との通信回数を記憶する記憶手段を有し、

前記通信時間算出手段は、前記記憶手段に記憶されている通信回数と、予め定められている各実行器との間の通信時間係数とから前記プログラムと前記各実行器との間の通信時間を算出することを特徴とする請求項１記載の計算機システム。

【請求項４】　前記通信時間算出手段は、全ての実行器または前記制御器から指示された処理能力に余裕のある実行器との間で通信を行い、そのときの通信時間から前記プログラムと前記各実行器との間の通信時間係数を定めることを特徴とする請求項１記載の計算機システム。

【請求項５】　前記制御器は、処理能力に余裕のある実行器を、プログラムの移動を希望する実行器に通知する通知手段を有することを特徴とする請求項１記載の計算機システム。

【請求項６】　複数のプログラムをそれぞれ実行する複数の実行器と、該複数の実行器を制御する制御器とからなる計算機システムの負荷分散方法において、

実行すべきプログラムと各実行器との間の通信量を記憶し、

該記憶した通信量に基づいて、前記プログラムを自己の実行器で実行した場合の該プログラムと他の実行器との間の移動前通信時間と、該プログラムを処理能力に余裕のある他の実行器に移動させた場合の該プログラムと移動先実行器以外の実行器との間の移動後通信時間とを算出し、

前記移動後通信時間が移動前通信時間より短くなるプログラムと実行器との組み合わせを抽出し、該抽出した実行器に該プログラムを割り当てることを特徴とする計算機システムの負荷分散方法。

2

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、ネットワークで接続された複数の計算機の負荷を適正に分散する負荷分散方法及びその負荷分散方法を適用した計算機システムに関する。

【０００２】

【従来の技術】複数の計算機がネットワークで接続されたシステムでは、システム全体の処理時間を短くするため各計算機の負荷分担を適正化することが要求される。

【０００３】負荷の適正化の方法として、各計算機の負荷が平均化されるようにする方法、あるいは負荷の最も軽い計算機に新たな仕事を分担させる方法等が考えられている。

【０００４】

【発明が解決しようとする課題】ところで、上述した負荷分散方法では、プログラムを負荷の軽い計算機に割り当てるためにはプログラムをその計算機に転送する必要があり、計算機間のデータの転送時間がプログラムの実行時間に比較して長い場合には、単純に負荷を平均化しただけでは計算機システム全体の処理効率を向上させることはできない。特に並列計算機システムでは、計算機の計算時間に比べて計算機間の通信時間が長いので、単に負荷を均等に割り当てたのでは計算機システム全体の効率を向上させることができないという問題点があった。

【０００５】本発明の課題は、計算機システム全体の処理効率を向上させることである。

【０００６】

【課題を解決するための手段】図１は、本発明の負荷分散方法を適用した計算機システムの原理ブロック図である。複数のプログラムをそれぞれ実行する複数の実行器と、該複数の実行器を制御する制御器とからなる計算機システムにおいて、実行器は、通信量見積り手段と、通信時間算出手段と、負荷割り当て手段とを有している。

【０００７】通信量見積り手段は、実行すべきプログラムと自己の実行器を含む各実行器との間の通信量を見積もる。この通信量見積り手段は、例えば実行すべき複数のプログラムと自己の実行器を含む各実行器と間の過去の通信量（通信回数、データ量等）を記憶し、その記憶してある通信量をプログラムと各実行器との間の通信量とする。

【０００８】通信時間算出手段は、通信量見積り手段に

よって見積もられた通信量に基づいて、プログラムを自己の実行器で実行した場合のそのプログラムと他の実行器との間の移動前通信時間と、そのプログラムを他の実行器に移動させた場合のプログラムと移動先実行器以外の実行器との間の移動後通信時間とを算出する。

【０００９】この通信時間算出手段は、例えばプログラムの実行に先立って他の実行器との間で通信を行い、そのときの通信時間を各実行器の通信時間係数として求め、通信量見積り手段により見積もられた通信量にそれぞれの実行器の通信時間係数を乗算して、プログラムと各実行器との間の通信時間を算出する。

【００１０】負荷割り当て手段は、通信時間算出手段によって算出された移動後通信時間が移動前通信時間より短くなるプログラムと実行器との組み合わせを抽出し、その抽出した実行器にプログラムを割り当てるように制御器に通知する。

【００１１】

【作用】本発明の計算機システムでは、例えば、それぞれのプログラムと各実行器との間の通信量を記憶しておいて、あるプログラムを自己の実行器で実行した場合の各実行器との間の移動前通信時間と、そのプログラムを他の実行器に移動させた場合の各実行器との間の移動後通信時間とを求め、移動後通信時間が移動前通信時間より短くなるようにプログラムを他の実行器に移動させるようにしたので、実行器間の通信時間が実行器の計算時間より長い場合にも、負荷を適正に分散して、計算機システム全体の処理時間を短縮することができる。

【００１２】

【実施例】以下、本発明の実施例を図面を参照しながら説明する。図２は、本発明の実施例の計算機システムの構成図である。この実施例の計算機システムは、ネットワークで接続されたＵＮＩＸ環境上に構築されている。ＵＮＩＸ環境では、複数のプロセスが１つの計算機上で動作するので、計算機上で動作するこれらのプロセスを実行器とし、複数の実行器を制御する制御用プロセスを制御器としてもよいし、プロセッサエレメント（ＰＥ）とメモリ等からなるハードウェアで実行器及び制御器を構成してもよい。

【００１３】計算機１は、実行器１１と制御器１４とを有し、計算機２は実行器１２を、計算機３は実行器１３をそれぞれ有している。特定の実行器１１から他の実行器にプログラムを移動させる場合には、実行器１１が移動させるプログラムを制御器１４に送信し、制御器１４がそのプログラムを処理能力に余裕のある実行器に送信するようになっている。なお、計算機１～３はバスを介して接続されている。

【００１４】各実行器１１～１３は、複数のプログラムを時分割で実行することができるとともに、制御器１４から処理能力に余裕のある実行器のリストを受け取る。各実行器１１～１３は、実行しようとするプログラムの

数が自己の実行可能なプログラム数より多いときには、そのことを制御器１４に通知する。そして、制御器１４から処理能力に余裕のある実行器のリストが通知されたなら、実行器１１～１３は、停止させたプログラムあるいはこれから実行しようとするプログラムを再開あるいは開始させるための情報を制御器１４に送る。

【００１５】また、実行器１１～１３は、制御器１４からプログラムを再開あるいは開始するための情報を受け取り、そのプログラムを再開する。制御器１４は、プログラムの転送要求を実行器から受け取ると、処理能力に余裕のある実行器のリストを転送要求のあった実行器に送る。また。制御器１４は、実行器から送られてくるプログラムを再開するたの情報を受け取り、それらの情報をプログラムの移動先の実行器に送る。

【００１６】次に、図３は、実行器１１の構成図であり、他の実行器１２、１３も図３と同様な構成を有する。同図に示すように実行器１１は、プロセッサエレメント（ＰＥ）２１と、後述する通信時間見積りプログラム、通信量見積りプログラム等を記憶するメモリ２２と、実行器１１が処理可能なプログラム数の上限値を記憶する上限値メモリ２３と、実行器１１が実行する各プログラムと自己及び他の実行器との間の通信回数を記憶する通信回数テーブル２４と、プログラムの実行順序が記憶されるキュー２５とを有する。

【００１７】例えば、通信回数テーブル２４には、プログラムａと実行器１１とのそれまでの通信回数として「３」、実行器１２との通信回数として「２」、実行器１３との通信回数として「１」が記憶されている。他のプログラムについても同様に各実行器との間の現在までの通信回数が記憶されている。

【００１８】次に、実行器１１と制御器１４の動作を図４及び図５のフローチャートを参照して説明する。以下、実行器１１を例にとり説明する。実行器１１は、図４のステップＳ１で実行すべきプログラムをキュー２５に登録する。そして、キュー２５からプログラムを取り出し順次実行する。そして、時間がきたなら、ステップＳ３でプログラムを停止させ、プログラムカウンタとデータとプログラム本体を保存する。そして、次のステップＳ４で現在実行しているプログラムとこれらか実行しようとするプログラムの数を調べる。このとき、実行すべきプログラムの数が自分が実行できるプログラム数を超えているときには、ステップＳ５で実行すべきプログラム数が自己の処理可能なプログラム数を超えていることを制御器１４に通知する。

【００１９】そして、制御器１４からその時点で処理能力に余裕がある実行器のリストが送られてきたなら、ステップＳ６で現在実行中のプログラムを停止させ、そのプログラムの移動先の実行器を指定して、プログラムカウンタの値と、データと、プログラム本体とを制御器１４に送り、キュー２５からそのプログラムを削除する。

このとき、実行器１１は、通信回数テーブル２４に記憶されている通信回数から算出される各プログラムと各実行器との間の通信時間と、各プログラムを他の実行器に移動させた場合の移動後通信時間とを算出し、移動後通信時間が移動前通信時間より短くなるプログラムと実行器との組み合わせがあったなら、その実行器をプログラムの移動先として制御器１４に指示する。

【００２０】他方、制御器１４から再開すべきプログラムのプログラムカウンタの値と、データと、プログラム本体とが送られてきた場合には、ステップＳ７でそれらのデータ及びプログラムを内部のメモリ２２に格納すると共に、そのプログラムをキュー２５に登録する。

【００２１】制御器１４は、図５のステップＳ２１で実行器から送られくる命令を待ち、例えば実行器ｉから命令を受け取ったなら、ステップＳ２２でその命令が実行器ｉのプログラム数を知らせるものか否かを判別する。

【００２２】ステップＳ２２の判別で、受け取った命令が実行器ｉのプログラム数を知らせる命令であると判別されたときには、各実行器の状態を調べるために、先ずステップＳ２３で実行器１１が実行中のプログラム数を調べ、実行器１１が空いているか否か、すなわち実行器１１で新たなプログラムを実行可能か否かを判別する。この判別で実行器１１に空きがあると判別されたときには、ステップＳ２４に進み、命令を発行した実行器ｉに実行器１１の処理能力に余裕があることを通知する。

【００２３】ステップＳ２３の判別で実行器１１が空いていないと判別されたとき、またはステップＳ２４の次には、ステップＳ２５に進み実行器１２が空いているか否か、すなわち実行器１２で新たなプログラムを実行可能か否かを判別する。この判別で実行器１２に空きがあると判別されたときには、ステップＳ２６に進み、命令を発行した実行器ｉに実行器１２の処理能力に余裕があることを通知する。

【００２４】ステップＳ２５の判別で実行器１２が空いていないと判別されたとき、またはステップＳ２６の次には、ステップＳ２７に進み実行器１３が空いているか否か、すなわち実行器１３で新たなプログラムが実行可能か否かを判別する。この判別で実行器１３に空きがあると判別されたときには、ステップＳ２８に進み、命令を発行した実行器ｉに実行器１３の処理能力に余裕があることを通知する。

【００２５】上記の処理により、実行器ｉ側では、どの実行器が空いているかを知ることができるので、後述する通信時間見積り処理等により実行中のプログラムを空いている実行器に移動させた場合の通信時間を算出し、通信時間の短くなる実行器にプログラムを移動させることができる。

【００２６】ステップＳ２２の判別で実行器ｉから送られてきた命令がプログラム数を知らせる命令でなかったときには、ステップＳ２９に進み、プログラムを再開すべき実行器を指定する命令が実行器ｉから送られてきたか否かを判別する。この判別で、プログラムを再開すべき実行器を指定する命令であると判別されたときには、次のステップＳ３０で、実行器ｉから送られてきたプログラム情報を指定された実行器に送信する。

【００２７】次に、実行器ｉで実行中のプログラムの通信量を見積もる通信量見積り処理（通信量見積り手段に対応する）を図６（Ａ）、（Ｂ）、（Ｃ）のフローチャートを参照して説明する。

【００２８】先ず、図６（Ａ）のステップＳ４１で、実行器ｉが実行中のプログラムと自己の実行器を含む各実行器との間の通信回数が記憶される通信回数配列を通信回数テーブル２４に設け、その通信回数配列の全ての要素を「０」で初期化する。次に、同図（Ｂ）のステップＳ４２で、実行器ｉで実行中のプログラムが各実行器と通信を行う度に、通信回数配列の該当するプログラムの各要素の中で通信の相手先の実行器に対応する要素の値を「１」インクリメントする。上記の処理により、例えば、プログラムａが実行器１１と通信した場合には、図３の通信回数テーブル２４のプログラムａと実行器１１との要素の値が「３」から「４」に変更される。

【００２９】次に同図（Ｃ）のステップＳ４３で、移動候補のプログラム（これを、プログラムｍとする）とそのプログラムｍの移動先候補の実行器（これを、実行器ｎとする）と間の通信量を見積もる。具体的には、通信回数配列の移動候補のプログラムｍと移動先実行器ｎとの要素の値、すなわち通信回数を通信量として求める。

【００３０】そして、次のステップＳ４４で、通信回数配列におけるその移動候補のプログラムｍと各実行器との要素の値の総和から、上記のステップＳ４３で求めた移動先候補の実行器ｎの要素の値を減算し、減算結果に「１」を加えたものを、プログラムｍを実行器ｎへ移動させたときの移動後通信量として見積もる。すなわちプログラムｍと各実行器との間の通信量の総和からプログラムｍと実行器ｎとの間の通信時間を減算することで、プログラムｍを実行器ｎに移動させた後の、プログラムｍと実行器ｎ以外の他の実行器との間の通信時間を求める。また、プログラムｍの移動前通信量として、通信回数配列におけるプログラムｍと実行器ｉ以外の他の実行器との要素の値の総和を求める。

【００３１】なお、ステップＳ４４の処理で、プログラムｍを実行器ｎへ移動させた場合の移動後通信量を見積もる際に「１」を加算しているのは、プログラムｍを実行器ｎへ移動させるときの通信量を加えるためである。

【００３２】以上のようにしてプログラムｍを実行器ｎへ移動させた場合の移動後通信量及び移動前通信量とを見積もったなら、同様にしてプログラムｍを空いている他の実行器に移動させた場合の通信量を見積もる。

【００３３】さらに、１つの移動候補のプログラムに関する移動後及び移動前通信量を見積もったなら、実行中

の他のプログラムについて同様に通信量の見積りを行う。そして、各プログラムの通信量を求めたなら、次にそれらのプログラムの通信時間を算出する。以下、通信時間見積り処理（通信時間算出手段に対応する）を図7（A）、（B）のフローチャートを参照して説明する。

【0034】先ず、図7（A）のステップS51で、予め定められている、一定量の通信を行うときにかかる通信時間を通信時間係数として設定し、通信時間係数の初期化を行う。この実施例では、プログラムと実行器との間の通信時間が通信回数に比例するものとしているので、通信時間係数として「1」が設定される。

【0035】通信時間係数の初期化が終了したなら、図7（B）のステップS52で、前述した通信量見積り処理で求めた、移動候補のプログラムmを実行器nへ移動させた場合の通信量に通信時間係数を乗算して移動後通信時間を求める。本実施例では、通信量として通信回数を用い、各実行器の通信時間係数を「1」としているので、通信回数配列の各要素の通信回数がそのまま通信時間となる。なお、通信時間係数が実行器毎に異なる場合には、通信回数配列の各要素に、対応する実行器の通信時間係数を乗算して通信時間を求める。

【0036】そして、上記のようにして求めた移動後通信時間が、移動前通信時間より短くなるプログラムと実行器との組み合わせを抽出し、そのプログラムの実行を中断し、抽出した実行器をプログラムを再開すべき実行器として指定して、中断したプログラムのプログラムカウンタの値とデータとプログラム本体を制御器14に送信する。

【0037】制御器14は、実行器からプログラムを再開させるべき実行器を指定してプログラム情報が送られきたなら、受信したプログラム情報を再開すべき実行器に送信し、プログラムを再開させる。

【0038】次に、本発明の負荷分散方法を、CPUの処理速度に比べてCPU間の通信時間の方が長い計算機システムに適用した場合について説明する。今、1000個のCPUからなる計算機システムで、1つのCPU（実行器）が実行できるプログラム数の上限値が「900」、CPUにおける1つのプログラムの計算時間が「10」秒、1つのプログラムを他のCPUへ移動させる場合の移動時間が「300」秒で、全てのプログラムは1つのプログラムから生成されるものとする。

【0039】各CPUに均等にプログラムを割り当てる場合には、1000個のCPUにそれぞれ1つのプログラムを割り当てることとなるので、プログラムを生成したCPUから999個のCPUにプログラムを送信するための通信時間が「300×999秒」、計算時間が「10秒」であるので、計算全体にかかる時間は「300×999＋10＝299，710秒」となる。

【0040】これに対して、本発明の負荷分散方法を適用した場合、プログラムを他の実行器に移動させた場合の計算時間より、1つのCPUにプログラムを集中させて場合の計算時間の方が短くなるので、1つのCPUに900個のプログラムを割り当て、残りの100個のプログラムを他のCPUに移動させることになる。その場合、100個のCPUにプログラムを割り当てるための通信時間が「300×100秒」、計算時間が「10×900秒」であるので、計算全体にかかる時間は「10×900＋300×100＝39，000秒」となる。

【0041】従って、CPUの計算時間に比べてCPU間のデータの転送時間が長い場合には、各CPUに均等に負荷を割り当てる方法より、本発明の負荷分散方法の方が全体の計算時間を短縮することができる。

【0042】なお、上記実施例では、通信回数テーブル25に記憶されているプログラムの現在までの通信回数をもって通信量としているが、過去の一定期間の通信量をもって、そのプログラムの通信量としてもよい。また、通信回数ではなく、実際に送信されたデータ量をもって通信量としてもよい。

【0043】また、上記実施例では、通信時間係数を「1」として各実行器間の通信時間が等しい場合について説明したが、例えば、全てのプログラムの実行に先立って全ての実行器間で通信を行い、そのときの通信時間を各実行器の通信時間係数として用いてもよいし、移動すべきプログラムを抽出する際に、全ての実行器間、あるいはプログラムを移動させることのできる実行器との間で通信を行い、そのときの通信時間を実行器間の通信時間係数として用いてもよい。また、プログラムの行った過去全て、あるいは一定期間内の通信時間を通信量で割った値を通信を行った実行器間の通信時間係数として用いてもよい。あるいは、各実行器間の通信時間係数を予め個々に定めておいて、それらの値を通信時間係数として用いてもよいし、ネットワークにおける各計算機の接続状態に応じて定められた値を通信時間係数として用いても良い。

【0044】

【発明の効果】本発明によれば、実行器での計算時間に比べて、実行器間の通信時間が長い場合に、プログラムを他の実行器に移動させたときの移動後の通信時間と、移動前の通信時間とを比較し、移動後の通信時間が短くなるようにプログラムを他の実行器に移動させることで、計算機システム全体の処理時間を短縮することができる。

【図面の簡単な説明】

【図1】本発明の原理ブロック図である。

【図2】実施例の計算機システムの構成図である。

【図3】実行器の構成図である。

【図4】実行器の動作を示すフローチャートである。

【図5】制御器の動作を示すフローチャートである。

【図6】（A）、（B）、（C）は通信量見積り処理のフローチャートである。

9　　10

【図7】（A）、（B）、は通信時間見積り処理のフローチャートである。

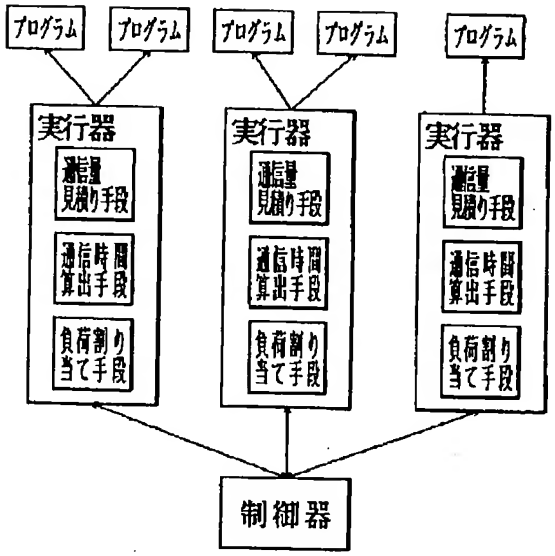
【符号の説明】

＊11～13　実行器
14　制御器

＊

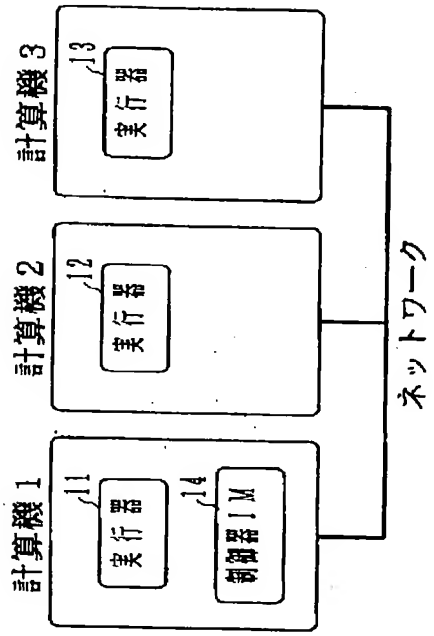
【図1】

本発明の原理ブロック図



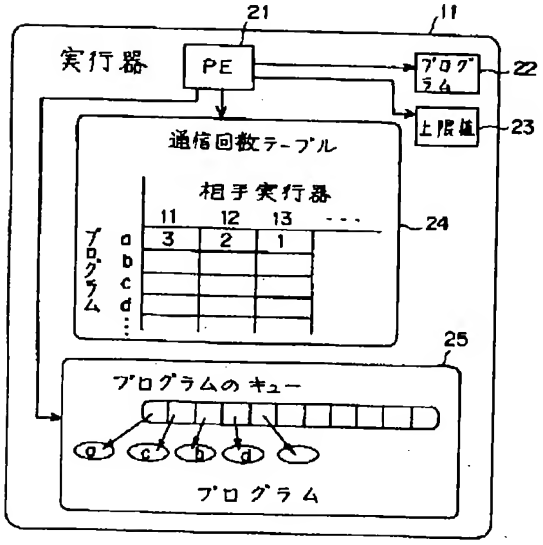
【図2】

実施例の計算機システムの構成図



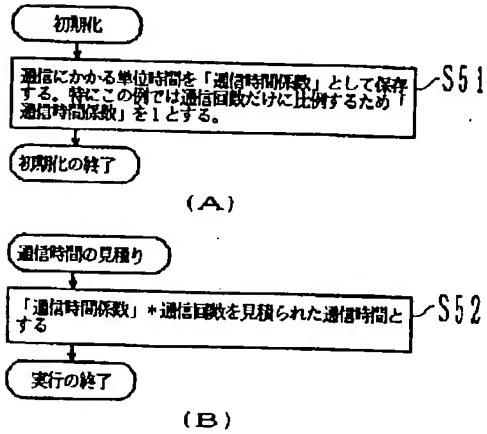
【図3】

実行器の構成図



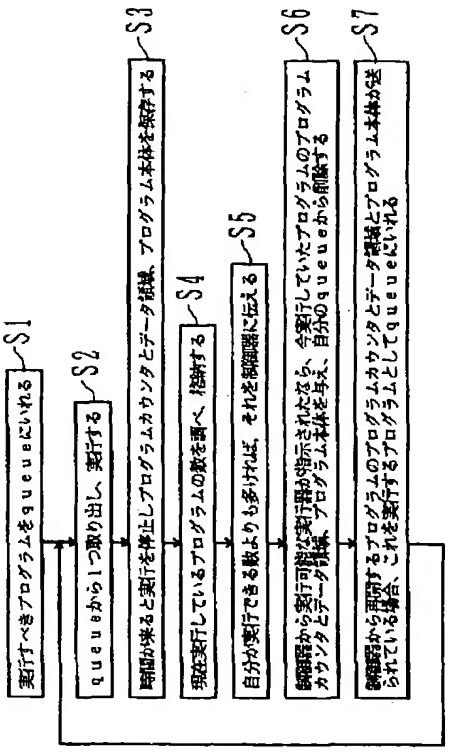
【図7】

通信時間見積り処理のフローチャート



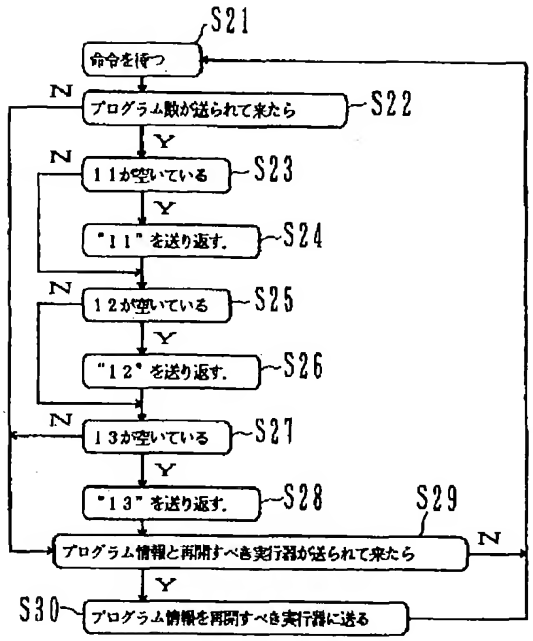
【図4】

実行器の動作を示すフローチャート



【図5】

制御器の動作を示すフローチャート



【図6】

通信量見積り処理のフローチャート